Learning Three-Dimensional Shape Models for Sketch Recognition

Leslie Pack Kaelbling, Tomás Lozano-Pérez
MIT Computer Science and Artificial Intelligence Laboratory

***Abstract*—Artifacts made by humans, such as items of furniture and houses, exhibit an enormous amount of variability in shape. In this paper, we concentrate on models of the shapes of objects that are made up of fixed collections of sub-parts whose dimensions and spatial arrangement exhibit variation. Our goals are: to learn these models from data and to use them for recognition. Our emphasis is on learning and recognition from three-dimensional data, to test the basic shape-modeling methodology. In this paper we also demonstrate how to use models learned in three dimensions for recognition of two-dimensional sketches of objects.**

***Index Terms*—sketch recognition, object recognition, computer vision**

I. Introduction

Object recognition is a fundamental required competence both for understanding human behavior and for generating sophisticated robotic behavior. A system should be able both to identify objects in a scene and to understand their relative placement in space. To support such broad applications, it is important for the recognition to be of object classes rather than particular object instances.

Artifacts made by humans, such as items of furniture and houses, exhibit an enormous amount of variability in shape. Unlike classes of natural shapes, such as animals, it isn't possible to smoothly morph between elements of the set of shapes of chairs. There is structure in the variability of these classes, however. Chairs have two arms or none; the base may be made up of legs or caster wheels, but that is independent of the shape of the back. This kind of structural variability may be appropriately captured in a probabilistic grammatical model, in which the terminals are primitive shape descriptions of basic parts of the object.

In this paper, we will concentrate on models of the shapes of objects that are made up of fixed collections of sub-parts. Our goals are: to learn these models from data and to use them for recognition. Our emphasis is on learning and recognition from three-dimensional data, to test the basic shape-modeling methodology. In this paper we will also demonstrate how to use models learned in three dimensions for recognition of two-dimensional sketches of objects.

This material is based upon work supported in part by the Singapore-MIT Alliance and in part by Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010 and DARPA Grant No. HR0011-04-1-0012.
Both authors are with the M.I.T. Computer Science and Artificial Intelligence Laboratory, 32 Vassar Street, Cambridge, MA 02139, USA; email: lpk@csail.mit.edu, tlp@csail.mit.edu.

The approach to object recognition based on 3D models, although popular in the past [1], [2], [3], is in contrast to most current work in object recognition, which is strongly image-based. There is an enormous amount of previous work in object recognition, which is impossible to review here even in barest outline. The closest recent work is the *constellation model* [4], in which objects are modeled by distributions over relations among salient image points in two dimensions.

A. Modeling an object instance

We will use rectangular boxes as the primitive shapes of parts in our models. Eventually, we may wish to use more sophisticated primitives, such as superquadrics [5], to model smoother, more complex shapes. However, our goal in modeling shapes is to capture the gross overall structure; if particular details (such as carving or precise curvature) are crucial for recognition, they might be better modeled as additional features such as texture in shape or intensity.

We can describe a simple chair, such as the one shown in the first pane of figure 2, using a set of six boxes. The dimensions of each box are described using three parameters. One box, in this case, the seat, is chosen as the global reference frame for the object. The pose (position and orientation) of each of the other boxes is described with respect to the base frame, using six parameters: three positions and three rotations. Thus, an instance of this model class is described with a total of 48 parameters.

There is a remaining issue of how, given a labeled collection of parts, to establish canonical relative reference frames. In the model, the frames are defined relative to other parts. So, for example, the z axis of the seat is defined to be pointing away from the legs and the x axis of the seat to be pointing away from the back. Once these two coordinates are defined, the third one may be computed.

B. Distributions over objects

Given the basic structural model, we can define a class of objects by defining a joint probability distribution over the model parameters. We will assume that the distribution is uni-modal; if we need to describe different *types* of chairs, we will do that with separate models.

Our strategy is to use a multivariate Gaussian distribution; but doing so is complicated by the fact that our model does not have the simple algebraic structure of $\mathbb{R}^n$. The relative positions of objects are real-valued, and so easy to model, but dimensions are strictly positive, and rotations

are elements of a special group. Following the approach of Fletcher et al. [6], we will define a joint Gaussian distribution over all the parameters, but in a space that is a product of specialized subspaces for each of the parameters. For the real-valued parameters, the space will simply be $\mathbb{R}$, but for the other parameter types, it will differ.

Dimensions: Since dimensions are elements of $\mathbb{R}^+$ (positive reals), an appropriate distribution for them is the lognormal. A variable X is lognormally distributed if $Y = \ln X$ is normally distributed. Because we want to define a joint distribution over all of our parameters, rather than putting a lognormal distribution on the dimension parameters, we will use ln to map them to a new space.

Rotations: A rotation can be represented in a variety of ways, including as three Euler angles, as a 3 x 3 rotation matrix, as a unit-vector axis and angle of rotation about that axis, and as a unit quaternion. The quaternion representation is particularly convenient because it lacks singularities [7]. Quaternion representations of rotations are points on the unit four-dimensional hypersphere.

There are two main approaches to representing a probability distribution on a hypersphere: one is based on generalizations of the von-Mises distribution for single angles, and the other is based on a "wrapped" Gaussian distribution. We will follow the wrapped Gaussian approach, because it will integrate effectively with Gaussian distributions on other parameters. This approach is very well described by Johnson [8].

A wrapped Gaussian distribution can be best understood first in the context of a single angle. We can represent angles as points on the unit circle (defined by $\langle \sin\theta, \cos\theta \rangle$). The mean of a wrapped Gaussian is an angle, μ. Once the mean is specified, we can make a line tangent to the circle at μ, as shown in figure 1, and put a univariate Gaussian distribution with variance σ^2 on that line. We can now take that tangent line and "wrap" it back around the circle. The probability density of an angle x is then defined to be

$$\Pr(x) = \sum_{i=-\infty}^{+\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2\sigma^2}(x - \mu - 2\pi i)^2) \ .$$

It is an infinite sum because the tails of the Gaussian continue to wrap around the circle infinitely. In practice, we assume the distribution is peaked and take only the first term of the series (and normalize the distribution accordingly).

The mechanism is essentially the same for the quaternion representation of rotations. The mean of the distribution is defined by a quaternion μ. Then, we construct an $\mathbb{R}^3$ space that is tangent to the unit four-dimensional hypersphere at μ.

Let us first consider the case in which μ is equal to the quaternion representing the identity rotation: $\langle 1, \langle 0,0,0 \rangle \rangle$. Then the mapping from a quaternion $q = \langle w, v \rangle$ into the tangent space at the identity is

$$\ln(q) = \frac{\arccos w}{\sin \arccos w} v \ ,$$

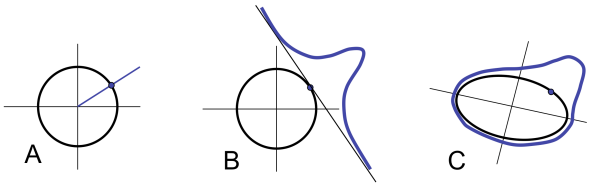

Fig. 1. A. Represent mean angle as point on unit circle. B. Construct tangent to circle at mean and center Gaussian distribution there. C. Wrap the tangent line, togther with the Gaussian, back around the circle.

which essentially is the scalar product of the axis of rotation with the angle of rotation about that axis. Now, we can place a Gaussian distribution on the tangent space, using a 3 x 3 covariance matrix Σ to describe the distribution.

When the mean μ is not equal to the identity, then to find the probability of a rotation represented by a quaternion q, we must first rotate the q by μ (this is accomplished by composing the conjugate of μ with q, $\mu^* q$), effectively setting μ to the identity and constructing a tangent space there, and then take the quaternion log of the resulting point. The tangent space should also be wrapped infinitely many times; we will just take the first term of the series, so

$$\Pr(q) \approx c \cdot \exp(-\frac{1}{2} \ln(\mu^* q)^T \Sigma^{-1} \ln(\mu^* q)) \ ,$$

where c is a normalizing constant.

Joint Gaussian: We can put all the pieces together to make a joint distribution over the whole parameter space. Consider a model that includes a single position p, a dimension d, and a rotation q. The mean of the distribution will be described by a vector $\langle \mu_p, \mu_d, \mu_q \rangle$, and the covariance will be described by a 5 x 5 matrix Σ. Note that q and μ_q are quaternions, with 4 components; but we only need three dimensions in Σ for the rotation, since we will have mapped the quaternion into an element of $\mathbb{R}^3$ when we consider covariance.

To compute $\Pr(\langle p, d, q \rangle)$ we begin by "subtracting" off the mean, to get a zero-centered vector:

$$\mathbf{X} = \langle p - \mu_p, \ln \frac{d}{\mu_d}, \ln(\mu_q^* q) \rangle \ .$$

Now $\mathbf{X}$ has a multivariate Gaussian distribution with mean $\mathbf{0}$ and covariance Σ. We will refer to distributions of this type, over positions, dimensions and quaternions as PDQ distributions.

II. Bayesian recognition

Assume we are given an "image", I, which is a collection of three-dimensional parts. We would like to classify collections of parts into different classes using a generative model. That is, for each class C_I, we compute $\Pr(C_i|I)$, then choose the C_i that maximizes that expression.

Using Bayes' rule, we have

$$\Pr(C_i|I) = \Pr(I|C_i)\Pr(C_i)/\Pr(I) \ .$$

Because we will compare these probabilities across different classes, but $\Pr(I)$ stays the same, we can ignore it. $\Pr(C_i)$ is a prior on getting elements of the different classes. So, we will concentrate on computing

$$\Pr(I|C_i)$$

for each class.

Let Θ_i be the parameters of the (PDQ) distribution of the shapes in class C_i. The full Bayesian solution would construct a distribution over possible parameterizations of the class, $\Pr(\Theta_i|C_i)$, yielding

$$\begin{aligned}\Pr(I|C_i) &= \int_{\Theta_i} \Pr(I|C_i,\Theta_i)\Pr(\Theta_i|C_i) \\ &= \int_{\Theta_i} \Pr(I|\Theta_i)\Pr(\Theta_i|C_i)\end{aligned}$$

This formulation is computationally intractable in general. Instead, we will assume that we have a single parameterization of the class (obtained via maximum likelihood estimation), Θ_i, so that

$$\Pr(I|C_i) \approx \Pr(I|\Theta_i) \ .$$

Both our images and our models (encoded by Θ_i), are made up of parts. In order to evaluate the probability $\Pr(I|C_i)$ we need to known an assignment of parts of the input "image" to parts of the model. We let A be a vector of n variables, one for each primitive component of the model, and let an $a \in A$ be a vector of indices of primitive components in the image, which correspond to the model components. Then,

$$\begin{aligned}\Pr(I|C_i) &\approx \Pr(I|\Theta_i,a) = \sum_a \Pr(I|\Theta_i,a)\Pr(a) \\ &\approx \max_a \Pr(I|\Theta_i,a) \ .\end{aligned}$$

For recognition purposes, we won't know the assignment of the parts in the image to those in the model, so we treat them as hidden variables. In principle, we should sum the probabilities over all possible assignments. Instead, we will pick the assignment that maximizes the probability, as computed by the joint Gaussian model presented above.

III. Learning

Given a collection of instances of a particular shape class, such as a chair, we need to be able to learn the parameters of the model. In this paper, we assume that the training examples are labeled; so an example is a set of boxes, which are labeled (seat, back, left-front-leg, etc.). Ultimately, this assumption will have to be relaxed, and an additional level of optimization will be required to maximize model likelihood over part-labelings.

With labeled parts, it is easy to convert a set of boxes into a vector of positions, dimensions, and quaternions of the type on which our model distribution is to be defined. Then our job is to estimate the model parameters given a set of such vectors.

A. Parameter Estimation

Estimating the means of positions is completely straightforward. For dimensions, we first take the log and then compute the sample mean. For rotations, it is more complicated.

For quaternions, the mean is the point on the four-dimensional unit hypersphere that minimizes the sum of the distances *on the hypersphere* to the data points. One difficulty with the quaternion representation for rotations is that it is redundant: a quaternion and its negation are both representations for the same rotation. When we measure the distance between two quaternions, we need to choose the minimum distance achieved by negating one of the arguments. Thus, the sample mean $\hat{\mu}_q$ of a set of quaternions q_i is defined to be

$$\hat{\mu}_q = \arg\min_p \sum_i \min_{a \in \{q_i, -q_i\}} \| \ln(p^* a) \| \ .$$

Johnson [8] shows how to convert this into a simple eigenvector problem. We let $\mathbf{Q}$ be the 4 x N data matrix; then the maximal eigenvector of $\mathbf{Q}\mathbf{Q}^T$ is $\hat{\mu}_q$.

To compute the covariance matrix, we must first "hemispherize" the quaternion data, by selecting the element of $\{q_i, -q_i\}$ that is closest to $\hat{\mu}_q$ for each quaternion component in the model. Now, we map each data point into the tangent space at the mean, getting a data set in a vector space where the sample covariance matrix can be estimated as usual.

As a proof of concept, we conducted some simple experiments on model learning. We generated sets of sample objects from the following classes: chairs with wheels, chairs with legs, chairs with legs and arms, tables, benches, and benches with arms. The sample objects were described as labeled sets of boxes. To see how much data was necessary for effective learning, we trained each model type on sets of increasing sizes and computed the log-likelihood assigned to new test data by each of the models. The results, indicate, relatively unsurprisingly, that a model with diagonal covariance works best with small numbers of examples, but is soon outperformed by the full-covariance model. We also found that we can get good performance with as few as 50 training instances.

IV. Recognition from three-dimensional input

Once we have trained a set of models, we can use them for recognizing novel instances of the learned object classes in complex scenes. To illustrate the algorithm, we will begin by assuming that a "scene" is actually made up of three-dimensional boxes, each described by its dimensions and its pose with respect to a global reference frame.

An interpretation of a scene is an assignment of boxes to parts of object model instances. There may be multiple

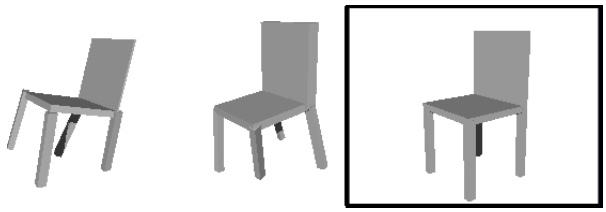

Fig. 2. Representative training examples and learned mean (in box) for two model classes.

instances of each object model in a scene. There may also be boxes in the scene that cannot be accounted for as part of an object instance; and there may be parts of object instances that are missing in the scene.

To arrive at a maximum-likelihood interpretation of a scene, it would be necessary to optimize the entire interpretation at once. Instead, we take a greedy approach, choosing a model class and looking for the highest-likelihood instance of it in the scene. If an instance with sufficiently high likelihood is found, it is added to the interpretation and the associated boxes are removed from the scene. It is necessary to provide a cutoff parameter for deciding whether a purported model instance is sufficiently highly likely.

A. Search

To find the highest likelihood instance of a given model in a scene, we use A^* search [9], in the space of partial matches between boxes in the scene and parts in the model. Each of the models has a "root" part with respect to which the coordinate frames of the other parts are defined. The initial interpretations are assignments of boxes in the scene to the root model part, at every possible orientation. The search proceeds by assigning boxes from the scene to parts in model, until a complete assignment is found or there is no further assignment that can be made without generating an instance with very low likelihood. This is a probabilistic extension of interpretation-tree search [10].

In order to guarantee that A* will find the highest likelihood interpretation, we must give it an admissible heuristic function to use to decide which of the current partial interpretations to expand first. A heuristic is admissible if it never underestimates the possible quality of a partial solution. In our case, we wish to find the complete assignment with the highest likelihood. An admissible heuristic must be optimistic about this value, so an obvious choice is the log likelihood of the complete model with the rest of its parameters filled in so as to maximize the conditional likelihood given the parameter values that have already been filled in.

More formally, let $\mathbf{x}$ be an interpretation that currently has parameters $x_1, \ldots, x_k$ filled in, and parameters $x_{k+1}, \ldots, x_n$ unassigned. Then the heuristic value for x is

$$H(\mathbf{x}) = \max_{x_{k+1},\ldots,x_n} \ln \Pr(\mathbf{x}|C_i)$$

$$= \max_{x_{k+1},\ldots,x_n} \ln \Pr(x_{k+1}, \ldots, x_n | x_1, \ldots, x_k, C_i) \ , \quad (1)$$

where C_i is the class model for which we are trying to find an interpretation. Since our probability model is just a multivariate Gaussian (in the tangent space), it is relatively straightfoward to compute H [11]. We can partition the covariance matrix as

$$\Sigma = \begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} \ ,$$

where s_{11} is the block of the covariance matrix relating the variables $x_1, \ldots, x_k$ to themselves, s_{12} relates the variables $x_1, \ldots, x_k$ to $x_{k+1}, \ldots, x_n$, etc. Then $\Pr(x_{k+1}, \ldots, x_n | x_1, \ldots, x_k)$ is itself a Gaussian with mean equal to $\mu_{k+1,\ldots,n} + s_{21} s_{11}^{-1} (x_{1,\ldots,k} - \mu_{1,\ldots,k})$, which maximizes the conditional likelihood.

The search proceeds by keeping an agenda, which is a set of partial interpretations. It is initialized with hypotheses that consider each of the boxes in the input to be the base part of the model, in each of 24 possible orientations that align faces in input to those of the corresponding part in the model. Then, on each step, it finds the $\mathbf{x}$ with maximal $H(\mathbf{x})$ and expands it by choosing a part in the model and considering extensions of $\mathbf{x}$ in which the part is matched to every remaining box in the scene, and putting all of the new interpretations into the agenda. It should, technically, generate 24 additional hypotheses, matching the new box to the selected part in every possible orientation; instead, we commit at this point to the single orientation with the best likelihood. If we considered all the orientations, this method would be guaranteed to find the maximum likelihood interpretation of M in the scene. The local commitment to an orientation voids this guarantee but substantially reduces the branching factor of the search. The remaining branching factor is still large, but the heuristic is quite powerful and tends to prune the search space effectively.

Optimism penalty: Although using the conditional mean to fill in the missing paramters is guaranteed to give us an overestimate of the likelihood of a partial hypotheses, it is very over-optimistic, and has the property of causing the search to systematically prefer hypotheses with only a few parts assigned. This is because any actual part will be somewhat worse than the mean, and therefore contribute a lower likelihood term.

So, we would like to be slightly less optimistic about as-yet unmatched parts. Consider a random variable X, normally distributed with mean μ_X and variance σ_X^2. If we want to be optimistic about the value this variable might take on, we should assume that it will be the mean value, and so an upper bound on the likelihood would be $\Pr(X = \mu_X) = 1/\sqrt{2\pi}\sigma$. Instead, a more realistic estimate of the likelihood of a hypothesis when a value for X is filled in might be the expected likelihood of a random value drawn from X, which is

$$E_X[\Pr(X = x)] = \int \Pr(X = x)^2 dx$$

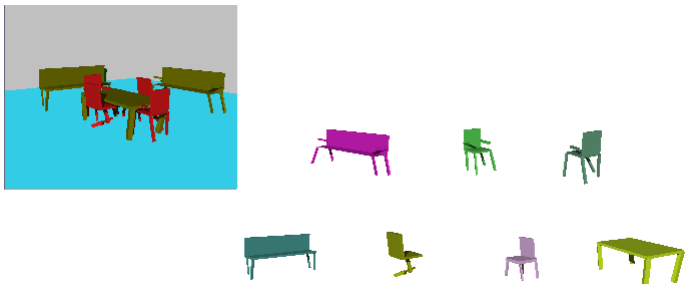

Fig. 3. Recognition algorithm interprets complex scene.

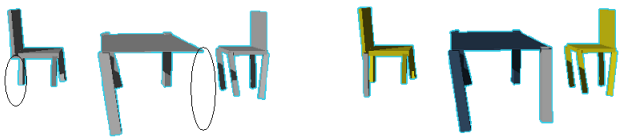

Fig. 4. Prediction of location and shape of missing parts (indicated with ellipses in first part of figure).

$$= \int \frac{1}{2\pi\sigma^2} e^{-\frac{1}{\sigma^2}(x-\mu)^2} dx$$
$$= \frac{1}{2\sqrt{\pi}\sigma} \int \frac{1}{\sqrt{\pi}\sigma} e^{-\frac{1}{\sigma^2}(x-\mu)^2} dx$$
$$= \frac{1}{2\sqrt{\pi}\sigma} \ .$$

We can see that the expected probability of a random draw from a Gaussian random variable is smaller than the probability of the mean, by a factor of $1/\sqrt{2}$.

In the p-dimensional multivariate Gaussian case, by similar reasoning, we find that the probability of the mean is

$$\Pr(X = \mu) = \frac{1}{\sqrt{(2\pi)^p |\Sigma|}},$$

and that the expected probability of a random draw is

$$\Pr(X = \mu) = \frac{1}{2^p \sqrt{\pi^p |\Sigma|}}.$$

So, we add an additional multiplicative penalty to an object with p as-yet unmatched parts of $1/\sqrt{2^p}$. Since our heuristic is in log-likelihood space, we subtract $p \ln 2/2$ from the conditional likelihood defined in equation 1, yielding (for $p = n - k$) a new heuristic:

$$H_2(\mathbf{x}) = \max_{x_{k+1},\ldots,x_n} \ln \Pr(\mathbf{x}|C_i) - (n-k)\ln 2/2 \ . \qquad (2)$$

B. Results

Figure 3 shows the operation of the recognition algorithm in a scene made up of multiple instances of different object classes. It quickly extracts the objects; although the branching factor of the search is large, the heuristic is powerful enough that very few extraneous paths are explored.

It is also possible to recognize objects with missing parts by allowing some parts to remain unmatched. A score can be readily computed for these partial hypotheses. However, more complete hypotheses above the acceptable likelihood threshold are preferred to less complete ones. In the first part of figure 4 is a scene in which a chair leg and a table leg are missing (indicated by ellipses). Nevertheless, the recognition algorithm is able to detect them. In addition, it is straightforward to predict the position and dimensions of the missing parts as shown in the second part of figure 4. This ability will be important when processing is being done on real images, in which the bottom-up segmentation process may have missed some parts. In such cases, we can recognize a partial view of an object, predict where the missing parts ought to be, and go back to verify their presence in the image.

V. Recognition from Two-Dimensional Input

Ultimately, of course, our goal is to do both learning and recognition from two-dimensional data. We will take a first step here by showing one way to use our learned three-dimensional models to do recognition from two-dimensional sketches, made up of projected faces of boxes.

There are a variety of possible strategies for doing two-dimensional recognition given a three-dimensional model. We will pursue one here that is a direct extension of the recognition we are doing from three-dimensional data. We will perform a "bottom-up" two-to-three dimension pass, in which we postulate the existence of three-dimensional boxes based on groups of four input points. This pass will postulate many more boxes than could possibly exist; but the recognition process is guided by the search heuristics and can ignore most of the incorrect hypotheses.

A. From two to three dimensions

Our goal is, given a set of points in the input, to generate all of the three-dimensional boxes that are consistent with those points being vertices of the box. For simplicity, we will start by assuming weak perspective (orthographic plus scale) projection. A simpler version of our problem has been solved in the vision literature: given three points on a rigid object, solve for the three-dimensional pose of the object [12]. This can be done reasonably straightforwardly, and the pose can be recovered, except for displacement along the viewing direction, about which no information is available in weak perspective projection. Our problem is somewhat more difficult, because our objects are not rigid; in particular, their dimensions are variable.

We will assume that the observed points arise from the projection of a face of the part box. Although seeing a single face of each part gives very weak information about that part, a single-face view of each of the parts of a complex object will typically provide sufficient information for recognition. Given a single face, we will not be able to

solve directly for the heights of all points in the image and dimensions of the corresponding box, we will only be able to derive only constraints on them. During the recognition process, we will have to optimize over the variables that remain free.

The orthographic projection of a rectangle in 3-space is always a parallelogram, and so there are really only three observational degrees of freedom: essentially the lengths of two sides, and one of the diagonals. In figure 5 we can see vertices labeled $v_0 \ldots v_4$.

Define h_i to be the "height," or distance orthogonal to the viewing plane, from vertex v_i to the viewer. We are using weak perspective projection, so these heights cannot be determined globally; we will arbitrarily set h_0 to 0, and constrain the remaining heights with respect to h_0. Finally, let s_x, s_y, s_z be the dimensions of the object. We have six unknowns: the three dimensions of the object and the relative heights of three of the vertices.

For the orthographic projection of a rectangular face, we can show that $h_2 = h_1 + h_3$. So we have two unknown heights. Further, we have no information at all about one of the dimensions (without loss of generality, let it be s_z) and so we will ignore it completely in the following and leave it as a free parameter to be optimized later. We have three measured variables (d_{01}, d_{02}, d_{12}) and four unknowns (h_1, h_3, s_x, s_y), which leaves us no choice but to treat one of the unknowns as a parameter. So, we will treat h_1 as a parameter, obtaining the following expressions for the other unknown quantities:

$$h_3 = \frac{d_{01}^2 - d_{02}^2 - d_{12}^2}{2h_1}$$
$$s_x = \sqrt{d_{01}^2 + h_1^2}$$
$$s_y = \sqrt{d_{12}^2 + h_3^2} \ ,$$

where the d_{ij} are distances between input points in the image. Note that there are two solutions, for positive and negative choice of h_1.

These equations can be degenerate when $d_{01}^2 - d_{02}^2 - d_{12}^2 = 0$, in which case the perceived figure is a rectangle. In that case, there are two possible interpretations.

In the first, $h_1 = 0$, so the edge v_0, v_1 is in the viewing plane. Then $s_x = d_{01}$, and we let h_3 be the free parameter. In the second, $h_3 = 0$, so the edge v_0, v_3 is in the viewing plane. Then $s_y = d_{03} = d_{12}$ and h_1 is the free parameter.

In practice, because of measurement error or inaccuracies in hand-drawn sketches, these constraints cannot be relied on. We have found that, it is more reliable to make an initial guess at the box parameters and use optimization of likelihood to choose their actual values, as described in the next section.

B. Recognition

Given a set of rectangular input faces, we can recover a set of possible three-dimensional boxes, which are missing depth information along the viewing axis and some

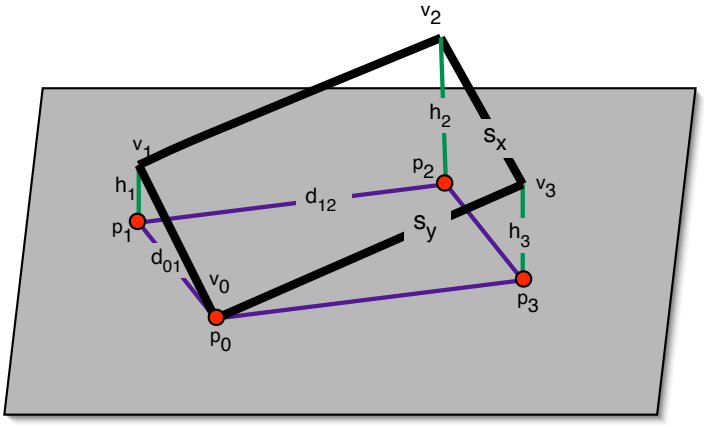

Fig. 5. Orthographic projection of face vertices onto the viewing plane.

dimension information. The goal remains to find the maximum likelihood interpretation of this scene given a three-dimensional model. Let I^2 be the two-dimensional "image" that is our input and let I be the three-dimensional scene that is our interpretation of the input image. Now we would like to compute

$$\Pr(I^2|C_i) = \int_I \Pr(I^2|I, C_i) \Pr(I|C_i) \ .$$

Given I, the random variables I^2 and C_i are independent, so we have

$$\Pr(I^2|C_i) = \int_I \Pr(I^2|I) \Pr(I|C_i) \ .$$

Performing the integral is intractable, so we will again approximate by finding the maximizing I:

$$\Pr(I^2|C_i) \approx \max_I \Pr(I^2|I) \Pr(I|C_i) \ .$$

We have already described $\Pr(I|C_i)$ as well as strategies for three dimensional recognition based on it. So, what is $\Pr(I^2|I)$? It is a "sensor" model, describing how three-dimensional objects are manifest in two-dimensional scenes. If the two-dimensional input were to come from hand-drawn sketches, it would encode information about typical drawing methods and styles; if the two-dimensional input comes from vision algorithms, then it would encode information both about typical viewpoints and about the kinds of errors made by the image-processing algorithms that produced it. Detailed models of this sort are very difficult to obtain.

Instead, we will start by assuming that there is no noise in the two-dimensional image-generation process (we will relax this assumption later), and therefore that there is a subset of I^2 that is consistent with any given I, and that they all have equal likelihood β. That is,

$$\Pr(I^2|C_i) \approx \arg \max_{I' \text{ consistent with } I^2} \beta \Pr(I'|C_i) \ .$$

We can proceed with A^* search as before; but it becomes more complex to evaluate the quality of a partial interpretation. The difficulty is that the missing parameters are not in the same space as the model parameters (the missing z parameter along the view direction, for example, is not explicitly represented; the model represents a part's position in a frame that is described with respect to the root part of the model). Thus, we will not be able to compute the heuristic analytically, by choosing the maximum likelihood conditioned on the values of a subset of the parameters in the PDQ distribution.

Instead, to compute the heuristic value for a partial interpretation, we perform a gradient-based optimization over the missing parameters, in order to find the maximizing parameter assignment, and use the resulting likelihood as the value.

Let I^2 be made up of a set of groups of observed points, $\hat{P}_j$, each of which arises from a projected view of a rectangular box. Using the approach described in the previous section, each observed set of points $\hat{P}_j$ can be converted into a parameterized description of a three-dimensional box, $B(\hat{P}_j, \theta_j)$, which determines the box's dimensions and its position and orientation in the view coordinate frame, as a function of the input data and an assignment of its free parameters θ_j.

A partial hypothesis consists of a mapping from some subset of parts of the object to a subset of the observed boxes. Let us assume that m parts have been matched in the current hypotheses. We have a vector of free parameters $\Theta = \langle \theta_1 \ldots \theta_m \rangle$; when these parameters are set, then k parameters $x_1 \ldots x_k$ describing m three-dimensional boxes are determined. We will denote this mapping from the measured point and the free parameters Θ to the k box parameters as $B(\hat{P}, \Theta)$. Now we can specify the heuristic function as:

$$H_3(\hat{P}) = \max_{\Theta} \max_{x_{k+1},\ldots,x_n} \ln \Pr(B(\hat{P}, \Theta), x_{k+1}, \ldots, x_n | C_i) - (n-k) \ln 2/2 \quad (3)$$

The maximization over Θ is done via gradient ascent; within the gradient ascent loop, the maximization over the remaining parameters is done analytically by maximizing conditional likelihood.

Parameter initialization: In order for the gradient search to work at all, we need to start with good initial values for the parameters.

In the general case, we initialize the scale parameter to a value s that makes the scaled versions of the recovered dimensions $s \cdot s_x$, $s \cdot s_y$, and $s \cdot s_z$ as close as possible to the mean dimensions of all the parts being matched as well as possible, in the least squares sense.

Each part has a depth offset, z, along the viewing axis. We let the depth of the centroid of the seat be 0, and each other part's z is defined relative to the seat. These z values are initialized by first determining the expected (according to the learned PDQ distribution) position of the part's centroid in the object's coordinate frame, using the object's orientation to determine that part's centroid in the global viewing frame, and then returning the z coordinate.

We will additionally be optimizing over a height and a dimension parameter. To initialize, we use the expected dimensions of the parts and choose heights that would explain the observed (projected) dimensions. This may be quite far from the truth, since the actual dimensions may be far from the mean of the distribution, but is a reasonable starting point.

Expected face match: Given the number of free parameters in early, highly partial hypotheses, and the optimistic nature of the search heuristic, we find that the recognition search can be very slow, because by optimizing the free parameters, almost every initial hypothesis can be made to seem very plausible. The parameter optimization is fairly expensive, and so we wish to reduce the number of hypotheses that we optimize carefully.

Consider a situation in which we have a single input face matched to the seat of a chair model. Given the orientation chosen for the match, and the signs of the heights, we can predict the position, orientation, and size of the projections of the legs and back in the input. These predictions may be fairly inaccurate, depending on the quality of the parameter estimation so far, and the amount of variance in the model. Nevertheless, it ought to enable us to check, for example, whether the legs are sticking up (as predicted) or down.

We will develop a new heuristic function, H_4, in which some of the parameters describing the positions of the unmatched parts, $x_{k+1}, \ldots, x_n$, are set, independently for each part, based on the input face "nearest" the projection of the part in image.

The definition of "nearest" is a bit tricky. Given the class model, C_i, and at least one assigned part, we have a distribution on the centroid of each part. This distribution is Gaussian in the tangent space, but not in our three-dimensional space. To get the centroid of a face, we can add a random variable representing the box dimensions to the part centroid. We could then transform it into the view frame. Then, ideally, we would find the centroid of the input face that had the highest likelihood in this distribution. But because the transformed distribution is not Gaussian, this is too expensive, so instead we use the *expected* location of the centroid of a part's face in 3D. Then, we take each of the centroids of the faces in the input, and transform them into the object's frame, with the z parameter set in order to make it as close as possible to the part face centroid, roughly maximizing its likelihood over the free z parameter. We choose the combination of a part face and an input face that minimizes this distance.

From this assignment, we infer the position of the expected centroid of the part. From this, we estimate the expected position of the part's centroid by simply translating the part's centroid over, by the mean dimension of the object (this is a gross approximation; it would be better, but slower, to convolve with the dimension distribution). This process tells us an approximate position of each unmatched part which would correspond to some input face.

Note that it does not constrain the rotation or the dimension of the part. We can use this estimated position to set some of the unassigned parameters in the PDQ distribution. We will indicate these partially filled in parameters describing the unassigned parts as $F(x_{k+1}, \ldots, x_n)$ in the new heuristic:

$$H_4(\hat{P}) = \max_{\Theta} \max_{F(x_{k+1},\ldots,x_n)} \ln \Pr(B(\hat{P}, \Theta), F(x_{k+1}, \ldots, x_n))|C_i) \quad (4)$$

Note that given this, there is no need to assess an "optimism penalty".

Noisy input: Our solutions so far have assumed that the two-dimensional input points were exactly correct. Because we expect to get two-dimensional input either from a human-generated sketch or from image processing, there will certainly be noise in the locations of the input points.

Let P be the "true" input points; that is, the projections of the vertices of the boxes in the three-dimensional scene. And let $\hat{P}$ be the observed input points, corrupted by noise. Let us further assume that the noise process is independent of the object class. Then

$$\begin{aligned}
\Pr(\hat{P}|C_i) &= \int_P \Pr(P, \hat{P}|C_i) \\
&= \int_P \Pr(\hat{P}|P, C_i) \Pr(P|C_i) \\
&= \int_P \Pr(\hat{P}|P) \Pr(P|C_i) \\
&\approx \max_P \Pr(\hat{P}|P) \Pr(P|C_i) \ .
\end{aligned}$$

Again, it is too costly to integrate out P, so we use the maximum instead. Now, $\Pr(\hat{P}|P)$ is a noise model, and $\Pr(P|C_i)$ is the generative model of the "true" appearance of items in class C_i, as discussed in section B.

This analysis leads us to add the parameters P to the list of variables to optimize over when trying to find an interpretation of a collection of image points as a member of class C_i. We assume, for computational convenience, that the noise is Gaussian and independent for each point. Letting p_j be the i^th "true" input point and the $\hat{p}_j$ the corresponding measured point, we arrive at the final form of our heuristic function:

$$H_5(\hat{P}) = \max_{p_j} H_4(P) - \sum_j (p_j - \hat{p}_j)^2 \ . \quad (5)$$

C. Results and Discussion

We have tested the approach described above on a variety of sketches of office furniture. Figures 6, 7, and 8 show a sequence of sketches in the top row of each figure; the computed model instance for the sketch is shown below. Each sketch is made up of 6 "faces". Note that the faces are not even close to being proper projections of actual model faces and that the geometric relationships are quite variable. The system can cope robustly with this type of variation.

The strength of the system is in reconstructing the most likely three-dimensional interpretation of a sketch. In addition, in most cases we have tried, the likelihood of a sketch given the correct model is higher than those of incorrect models. But not always. The system as it currently stands is prone to false positives, especially when models are subsets of each other. For example, a very distorted chair could be interpreted as a table by ignoring the back. We expect that this can be improved with additional training data but this also suggests that a better approach to comparing interpretations of different subsets of the data might be required.

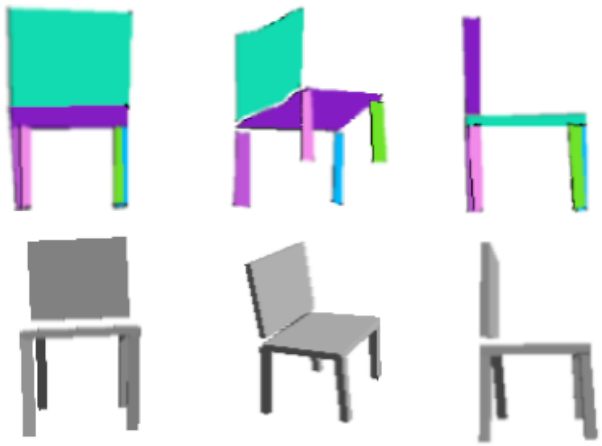

Fig. 6. Three-dimensional interpretations (bottom row) for chair sketches (top row).

We are currently pursuing a number of extensions of the approach described here. When dealing with a large number of possible models, one would like to avoid having to match the data against each model sequentially; we are pursuing rapid ways to retrieve plausible models based on relationships among observations. We plan to extend the approach to deal with actual images of objects, not just sketches; this requires a way of processing the images to identify potential matches to the object parts.

References

[1] W. Grimson, T. Lozano-Perez, and D. Huttenlocher, *Object Recognition by Computer: The Role of Geometric Constraints*. MIT Press, 1990.

[2] A. Pope, "Model-based object recognition: A survey of recent research," in *Univ. of British Columbia*, 1994.

[3] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Prentice Hall, 2001.

[4] R. Fergus, P. Perona, and A. Zisserman, "Object class recognition by unsupervised scale-invariant learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2003.

[5] A. Pentland, "Perceptual organization and the representation of natural form," *Artificial Intelligence*, vol. 28, pp. 293–331, 1986.

[6] P. T. Fletcher, S. Joshi, C. Lu, and S. Pizer, "Gaussian distributions on lie groups and their application to statistical shape analysis," in *Proceedings of Information Processing in Medical Imaging*, 2003, pp. 450–462.

[7] B. K. P. Horn, *Robot Vision*. The MIT Press, 1986.

[8] M. P. Johnson, "Exploiting quaternions to support expressive interactive character motion," Ph.D. dissertation, Massachusetts Institute of Technology, 2003.

[9] P. E. Hart, N. Nilsson, and B. Raphael, "A formal basis for the

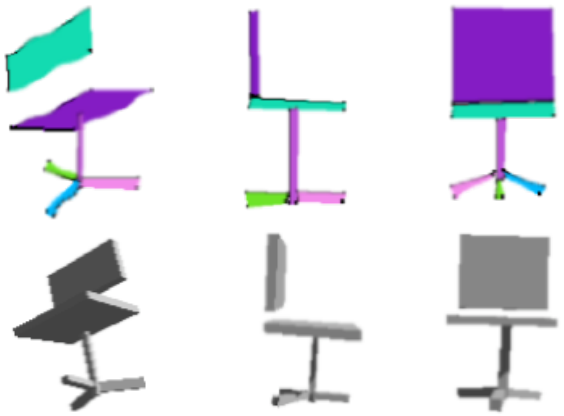

Fig. 7. Three-dimensional interpretations (bottom row) for office chair sketches (top row).

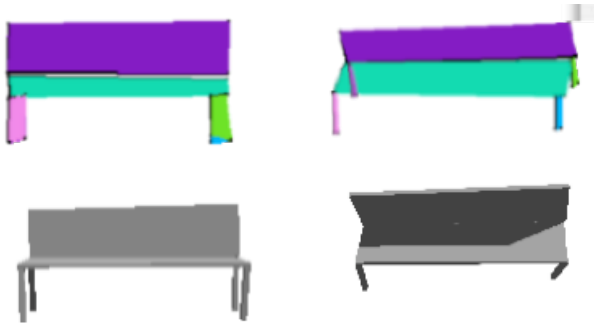

Fig. 8. Three-dimensional interpretations (bottom row) for bench sketches (top row).

heuristic determination of minimum cost paths." *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, 1968.

[10] W. Grimson and T. Lozano-Perez, "Localizing overlapping parts by searching the interpretation tree," *PAMI*, vol. 9, no. 4, pp. 469–482, July 1987.

[11] M. I. Jordan, *An Introduction to Probabilistic Graphical Models*. To Appear, 2004.

[12] T. Alter, "3d pose from three points using weak perspective," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, 1994.